Amendments to the Specification:

Please amend paragraph 0023 to read as follows:

[0023] The drives and their associated computer storage media, discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 110 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190, and video interface 190 and monitor 191 may be associated with a graphics interface 182, a graphics processing unit (GPU) 184, and video memory 186 in a generally known manner. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface [[190]] 195.

Please amend paragraph 0032 to read as follows:

[0032] XLANG/s is compatible with many Internet standards. XLANG/s is designed to use XML, XSLT ~~(http://www.w3.org/TR/xslt)~~, XPATH ~~(http://www.w3.org/TR/xpath)~~, XSD (XML Schema Definition) and WSDL (Web Services Description Language) as

supported standards and has embedded support for working with .NET based objects and messages. WSDL is described in a document titled "Web Services Description Language (WSDL) 1.1," W3C Note January 2001, by Microsoft and IBM Research, Copyright © 2000 Ariba, International Business Machines Corporation, Microsoft, and is hereby incorporated by reference in its entirety. The XLANG/s language is syntactically similar to C#, thus a C# specification may also be referenced as an aid to understanding the exact syntax. The semantics embodied in XLANG/s are a reflection of those defined in a document entitled "Business Process Execution Language for Web Services," Version 1.1, dated Mar. 31, 2003, published by Microsoft, IBM and BEA for the definition of Business Process semantics, which is also hereby incorporated by reference in its entirety. The Business Process Execution Language for Web Services specification is commonly referred to as the "BPEL4WS" specification. As may be appreciated, therefore, the use of XLANG/s is most advantageous when applied to a business process.

Please amend paragraph 0047 to read as follows:

[0047] Referring now to FIG. 3, a graph 300 illustrating resource management in a message processing system in accordance with one embodiment of the present invention is provided. In FIG. 3, X-axis 302 corresponds to a metric used to measure the workload of a message processing system such as, for example, memory occupied by operations, processor power in use and the like. X-axis 302 varies from a value of zero to a maximum value, denoted by "Max." "M" corresponds to a target utilization of the available memory and resources of the system. Y-axis 304 corresponds to the deviation between a given workload and the target utilization M. Accordingly, the value of the deviation is zero at the target utilization M.

Please amend paragraph 0062 to read as follows:

[0062] At step 510, a determination is made as to whether an instance in question is blocked. The determination of step 510 may be made either as a result of receiving a message, or may be performed on a periodic or other basis according to the dehydration

controller's 403 functionality. As discussed above, an instance may be blocked as a result of, for example, waiting to receive a message. Thus, it will be appreciated that in most embodiments, the blocked condition of an instance is not the result of receiving a message at step 505. If the instance is not blocked, then the method 500 processes the message (if a message was received) at step 535 according to the message system's usual protocols. Once the message has been processed at step 535, or if no message was received, the method 500 proceeds to step 545, which will be discussed in greater detail below.

Please amend paragraph 0072 to read as follows:

[0072] If the result of the determination of step 560 is the same as the determination of step 525 of Fig. 5A, namely that $t_{I(S)} \leq T$, then the instance *I(S)* is left in active memory as at step 570, as the processing and resource benefit from dehydrating the instance *I(S)* is not expected to outweigh the processing and resource overhead associated with dehydrating and rehydrating the instance *I(S)*. If, at step 560, the determination is made that $t_{I(S)} > T$, then the method 500 proceeds to step 565. At step 565, a request to dehydrate the instance *I(S)* is queued and ultimately the instance *I(S)* is dehydrated.